Spacecraft Pose Estimation via Monocular Image Processing: Dataset Generation and Validation

Michele Bechini[⋆†], Paolo Lunghi[⋆], and Michèle Lavagna[⋆]
⋆Politecnico di Milano, Aerospace Science and Technology Dept.
Via La Masa 34, 20156 Milano, Italy
michele.bechini@polimi.it · paolo.lunghi@polimi.it · michelle.lavagna@polimi.it
[†]Corresponding author

Abstract

A new tool capable of rendering high-quality spaceborne synthetic images via ray-tracing is here presented. The assumptions and simplifications used to model both the spacecraft and the background objects are reported, and the trade-off between accuracy and rendering time is discussed, leading to the default settings of the tool. The optical properties of a simplified spacecraft model have been tuned during the qualitative and quantitative validation phase, successfully carried out and detailed in the article. The steps involved in the creation of publicly available labeled spaceborne image datasets with the proposed tool are also provided and discussed in the paper.

1. Introduction

New classes of missions that envision a major role for autonomous close proximity operations, with a particular interest in non-cooperative artificial objects, like formation flying missions (FF) with fractionated scientific payloads, on-orbit servicing demonstrators (OOS), active debris removal included, gained increasing attention in the last years, aiming to perform regular in-orbit services.[14] Significant technology development is still needed to make these missions feasible. The proximity operations and maneuvering impose a high level of reactivity of the chaser with respect to the target leading to the need for guidance, navigation, and control (GNC) chains solved autonomously onboard, ensuring timeliness, reactivity, effectiveness, and robustness in both nominal and off-nominal operations. The first ring of that chain is the autonomous relative navigation, hence the estimation of the relative pose (position and attitude) of the chaser with respect to the target. Dealing with an uncooperative target represents the most challenging scenario for the relative pose estimation task since no information is shared between the chaser and the target (except for the a priori knowledge of the geometry in the case of a known target), reducing the possibilities to only those that exploit only the capabilities of the chaser. Among others, sensor suites which include monocular cameras are largely considered as one of the most attractive solutions to acquire measurements for the onboard GNC chain[23] and solutions including cameras operating in the visible spectrum have been widely studied and applied in the context of both cooperative and uncooperative rendezvous.[8, 19]

"Classical" solutions to deal with the pose estimation via monocular images of known uncooperative targets include the extraction of hand-crafted features of the target (corner,[34] edges,[7] etc.) from the image via dedicated image processing algorithms. The relative pose of the camera of the chaser with respect to the target is then retrieved by solving the Perspective-n-Points (PnP) problem between the 2D features extracted from the image and the features of a 3D CAD model of the target available onboard. Among the algorithms proposed, the Sharma-Ventura-D'Amico (SVD)[35] achieved state-of-the-art (SOTA) performances by exploiting the edges of the target extracted via Hough Transform.[13] More recently, Artificial Neural Networks (ANNs) revealed to be in many different contexts, a valuable tool to process the acquired images. The different "modern" approaches for relative pose estimation involving the use of ANNs, mostly Deep Convolutional Neural Networks (CNNs) can be classified into two branches: direct CNN regression,[32, 37] and PnP solvers aided by CNN.[9, 29] The former approach aims to recover the relative pose via directly processing the input image with a CNN (usually split into two branches) that outputs the relative position and the relative quaternion. Instead, the latter approach includes a first step in which the image is fed to a CNN that outputs the 2D keypoints that are subsequently fed to the PnP solver.

From the results of the 2019 ESA's Kelvins Pose Estimation Challenge[17] it can be concluded that the most effective approach to deal with relative pose estimation via monocular images is using CNNs aided PnP solvers. Despite that, the robustness and effectiveness of this approach rely on the parameters learned by the CNN during the training

phase. To achieve a proper training of the CNN, labeled image datasets are needed but, despite that, currently, there is a lack of publicly available spaceborne image datasets, strongly limiting the applicability of CNNs aided PnP solvers to a wide range of possible scenarios and acting as a bottleneck for the development of new algorithms. To overcome this limitation, a spaceborne image dataset generation tool is here proposed. The images generated by the tool have been successfully validated by taking the Spacecraft Pose Estimation Dataset (SPEED) Dataset[18] as reference. The tool developed has been used to generate fully-synthetic spaceborne noiseless image datasets using a simplified 3D CAD model of Tango from PRISMA mission[12] reconstructed from a previous work,[35] since the official CAD model is not publicly available. The datasets generated are made publicly available and comprise a Pose Estimation Dataset,[1] a dataset tailored to bounding box extraction and semantic segmentation,[2] and a dataset labelled with Tango edges.[3] The main contributions of the proposed work can be summarized as:

- Definition of a tool for spaceborne synthetic noiseless image dataset generation via physically-based ray-tracing that can be applied to a wide range of scenarios including different spacecraft models, space resident objects as background (both airless or with an atmosphere layer), and different camera models.

- Qualitative and quantitative validation of the images generated by the proposed tool using Tango as spacecraft and the SPEED dataset as reference.

- Generation of publicly available datasets that can be combined among them (through the different labels associated with the same images common to all the datasets) to generate a multi-purpose dataset that can be applied to a wide range of tasks to be exploited via training of CNNs.

Notice that the image noise level can be dependent from the application scenario (e.g. for spacecraft orbiting airless bodies, the electric field generated by the interaction between the plasma and the body can be not negligible[4] and this can increase the electronic noise in the images) thus, to increase the flexibility of the tool and since the noise can be easily added to the images in post-processing, the images generated by the tool are noiseless. The remainder of the paper is organized as follows: Sec. 2 reports an overview of the available software for image generation via ray-tracing and the already publicly available datasets; the tool proposed is then described in detail in Sec. 3, while the validation of the images generated is discussed in Sec. 4. An overview of the datasets generated and their labels is provided in Sec. 5. Conclusions resuming the main outcomes of this work and hints for possible future developments are finally reported in Sec. 6.

2. Related Works

Despite the high interest in spaceborne images, the availability of real spacecraft images taken during a space mission is very limited, hence there is the need to generate realistic images synthetically, offering also the possibility to have full control of the scene. Generating realistic synthetic images is a well-studied task that is usually achieved via ray-tracing. Ray-tracing is a rendering technique that relies on the evaluation and simulation of the path of view lines which starts from the observer camera and ends on generic virtual objects. This, along with the simulated light rays from the light sources to the virtual object, enables the calculation of the color intensity of the associated pixels. Using physically based ray-tracers, hence simulating the true physics of the light, offers the possibility to generate synthetic images with a high degree of accuracy.[36] Currently, there are several tools, mostly based on OpenGL or ray-tracing, that offer the possibility to create a 3D scene to be rendered. Among that, the commercial solutions available comprise PANGU,[27] firstly developed to create synthetic planetary surface images but improved with the possibility of generating also artificial objects, and Airbus SurRender,[6] which handles various space objects such as planets, asteroids, satellites, and spacecraft. Some well-known simulators, widely adopted for video games and animations, like Blender[10] and Unreal Engine[16] have been used to render space-related scenes but, despite the visually impressive scenes, these simulators lack the realism needed being mostly devoted to cope with human vision.[6] Among the open-source software, both POV-Ray (Persistence of Vision Raytracer)[31] and PBRT (Physically Based Ray-Tracing)[28] are the most promising, offering the possibility of directly accessing the code and eventually patching it to tailor the software to the user's needs.

Concerning the spaceborne synthetic image datasets, the only publicly available are the Unreal Rendered Spacecraft On-Orbit (URSO),[32] SPEED,[18] and SPEED+[26] (improved version of SPEED). URSO is a synthetic dataset of 5000 RGB images of both Soyuz and Dragon spacecraft models with Earth as background rendered with a simulator based on Unreal Engine 4 with a resolution of 1080×960 pixels. Game engines have been criticized for the lack of photometric accuracy of the camera sensors,[6] but the author of URSO claims that efforts have been put to implement physically-based shading models and cameras in such engines and that the custom simulator used to synthesize URSO obtains photorealistic images. Despite that, no validation of the images in URSO is provided. The SPEED dataset and

its improved version, the SPEED+, are used for the ESA's Kelvins Pose Estimation Challenges, with SPEED being the first publicly available dataset of spaceborne images ever released. SPEED is composed of 15000 synthetic grayscale images of Tango obtained using the Optical Simulator,[5] based on an OpenGL rendering pipeline, and 300 grayscale images obtained using a mock-up. The SPEED+ is composed of 60000 synthetic grayscale images of Tango generated as for SPEED, 6740 grayscale mock-up images obtained by introducing hardware in the loop to simulate the Earth albedo, and 2791 grayscale mock-up images with sun-lamps used to obtain effects like flares.[25] All the images have a resolution of 1900×1200 pixels and have been validated against real images of Tango from PRISMA mission employing histogram comparison.[17] Further details regarding SPEED and SPEED+ are available in the related articles[17,25] for the interested reader.

3. Dataset Generation Tool

For the dataset generation tool presented here, it has been evaluated the possibility of using PBRT as ray-tracing software on which to build the tool. The main advantage offered by PBRT is the possibility of simulating directly the full lens-set of a real camera[28] in a physically-based way, making it possible to directly model the full image acquisition device. Despite that, the main issue identified in PBRT is the simulation of the Earth's atmosphere and the scattering properties. The density of the atmosphere layer can be defined in PBRT only on a regular grid that is then interpolated to retrieve the scattering effects on the light rays, this leads to artifacts in the image as shown in Fig. 1 (notice the brighter regular dots, most evident over the seas), also for a resolution of the grid of about 1 Km. We acknowledge that the issue can be overcome by patching the available source code but, for the tool reported here, it has been preferred to use a ray-tracing software that more easily handle the atmosphere simulation, due to its major role in achieving realistic and representative images.

Figure 1: Example of artifacts in atmosphere layer.

Due to this issue and although PBRT offers in its latest release the possibility of performing ray-tracing via GPU acceleration, for the tool presented here it has been decided to use POV-Ray as ray-tracer, despite a few well-known limitations[15] (no GPU, long parsing time, etc.). POV-Ray has been already successfully employed to render via ray-tracing both spaceborne natural landscapes like the Moon's craters,[15,20] and spacecraft.[11,33] The remainder of this section is devoted to the modeling assumptions used to render SPEED-like images, with a particular focus on Earth and atmosphere modeling in Sec. 3.1, and on the spacecraft model adopted and the pipeline to feed the mesh to POV-Ray in Sec. 3.2. The POV-Ray version used for this tool is a patched version of POV-Ray 3.7 made available in[11] that restores the possibility to exclude the computation of the bounding box around each object, avoiding errors in the rendering of scenes with space-scale distances. It is here remarked that the assumptions introduced in the next sections can be easily generalized to different celestial bodies and target spacecraft by simply changing a few hyperparameters, but keeping the core of the tool unchanged, making it well suited to simulate several scenarios of interest (e.g. Mars and its atmosphere as background).

3.1 Earth and Atmosphere modeling

The first step in modeling a celestial body is to retrieve the shape of the body itself. Concerning objects like the planets of the Solar System and most of their moons, they can be firstly approximated as spheres. The unit of measure used

in the tool is the kilometer. To improve the quality of the image it is possible to recover the true ellipsoidal shape of the planet, by modeling it as a sphere and then applying a scale factor. The scale factor used to simulate the Earth's oblateness is equal to 0.9967 applied along the z-axis of the Earth-centered inertial (ECI) reference frame. The next step toward the rendering of realistic images is the definition of the textures. For airless bodies, like the Moon, the only step needed is to apply a realistic surface texture by using the option "*map_type 1*" (hence by directly attaching the 2D texture over a 3D sphere, with the relative latitude-dependent shrinking). The same process can be applied also to the more complex case of celestial bodies with atmosphere and clouds since, as for the Earth case, there exist textures that include both the terrain and the clouds map in a single image. This approach leads to good-quality images, but the fact that the clouds are stuck over certain regions of the Earth for all the images, the optical properties of clouds, seas, and terrains are all equal, and the fact that the diffusion of the sunlight through the atmosphere is not simulated, make this approach not well suited to build datasets for CNN training.

To overcome the issue related to the clouds, here it is used a clouds-only texture that is applied over a shell defined to have a random thickness between about 625 m and 1250 m, with a minimum altitude of 650 m and a maximum altitude of 3150 m. Since the cloud map is a grayscale image, with black associated with regions where there are no clouds and white where there are thick and dense clouds, it is possible to tune the parameters of the texture in POV-Ray such that the transparency of the texture is mapped following the density of the cloud layer. This makes it possible to have the Earth's surface still visible under low-density clouds, see Fig. 2 where the semi-transparent clouds are evident. A random rotation angle with respect to the z-axis of the ECI frame is defined at each generated image and applied to the cloud texture to make the clouds rotate with respect to the ground. Concerning the Earth's surface, it has been decided to include in the Earth's model, similarly to,[11, 15] a differentiation of the optical parameters of the lands with respect to the seas. This is accomplished by using a binary map that splits water and lands. Defining different optical properties (especially for the reflection over the seas) makes the images more photorealistic. To improve the accuracy of the images, it has been considered also the possibility to include the topography of the terrain. To deal with that there are two options available in POV-Ray:

- Bump-map approximation: the irregularity of the surface is emulated by modifying the surface normal vector, hence altering the way the light reflects off the surface. This approach generates irregularities (i.e. bumps) by only modifying the reflection of the light, the geometry is not affected, so the original shape is preserved.

- Isosurface function: the isosurface object makes it possible to mathematically describe the topography of the terrain on POV-Ray. In this case, the geometry of the object is truly modified according to a model defined by the user, hence the result is more accurate also in terms of shadows in the image.

Both the bump-map approximation and the isosurface can be defined starting from a grayscale texture with the altitude mapped into the intensity of each pixel. The output map is here rescaled to have a maximum altitude of about 9 Km over the sea level, corresponding to the altitude of Mount Everest. A comparison between of the same images rendered without topography (Fig. 2a), with bumpmap approximation (Fig. 2b), and with isosurface function (Fig. 2c) is reported in Fig. 2, where all the images have been overexposed to better appreciate the differences in the topography. From the images reported it can be noticed that excluding the topography lead to unrealistic images and poor quality. The image obtained using the bumpmap approximation has a higher quality, with mountains clearly visible (due to the correct shadowing). Notice that the transparency of the clouds with low density allows us to see the Earth's texture under them. The image achieved using the isosurface is almost identical to the image obtained with the bumpmap, except for the region close to the terminator, where the shadows are better evaluated and defined for the case of the isosurface. Notice that to better appreciate the effect of including the topography, the images in Fig. 2 are rendered without including the atmosphere.

From this, it is clear that to achieve a high level of photo-realism in the images, it is mandatory to include the topography in the image generation pipeline. This result holds for any celestial body used as background for spaceborne images and the improvement in the quality of the image is higher as the relative distance between the camera and the celestial body is reduced (e.g. for Low Earth Orbits). Despite the comparable results in terms of quality of the image previously discussed, it must be noticed that using the bumpmap approximation gives a computational burden almost null with respect to the case without topography, while by using the isosurface, the rendering time is strongly increased. Hence, due to the reduced computational time and the high quality of the output images, the bumpmap approximation is used as the default method in the presented tool.

The last part considered in the Earth model is the atmosphere. The aspect of the atmosphere and the color that it assumes (from dark blue to red close to the terminator) depend on the spectrum of the light emitted by the Sun (that here is modeled to have a white light), the density of the particles and their kind in the atmosphere, and the length of the path of the light through the atmosphere. Concerning the interaction between the particles and the light, it is well known that the incident light over particles is scattered in all directions following some given law. The interaction between

(a) Image rendered without topography.
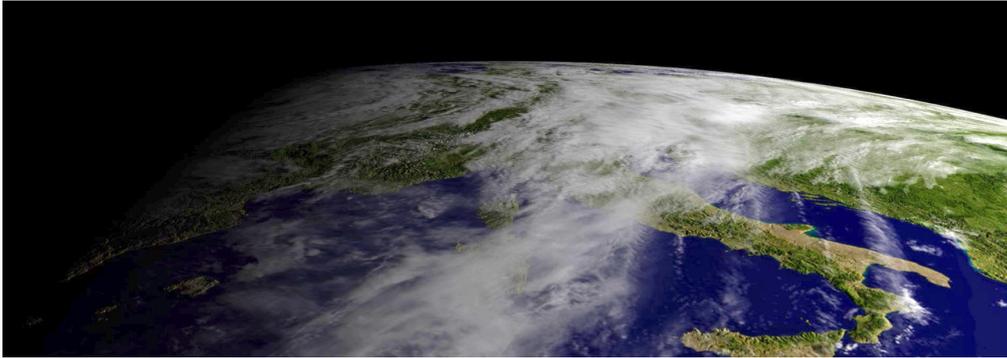
(b) Image rendered with bumpmap approximation.
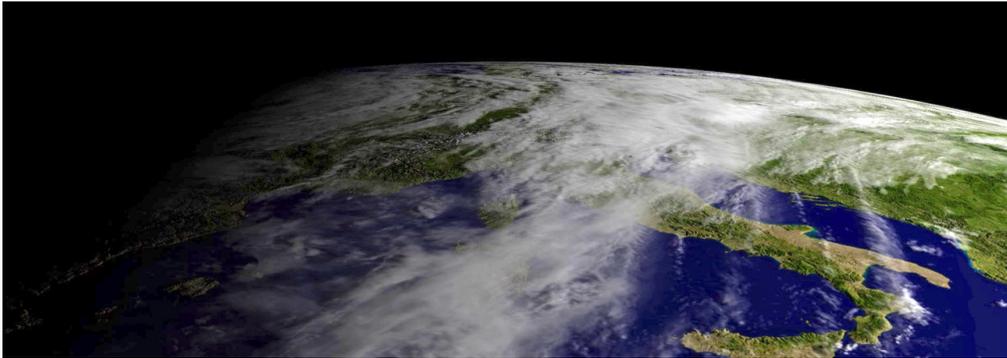
(c) Image rendered with isosurface function.

Figure 2: Comparison of images rendered with and without topography information.

the sunlight and the atmosphere particles can be well approximated by using the Rayleigh scattering. Following this model, the quantity of scattered light is dependent on the angle of the incident light. It is greater when the incident light is parallel or anti-parallel to the direction of observation and smaller when the incident light is perpendicular to the direction of observation. For a high-fidelity model, it must be also included that the Rayleigh scattering in the atmosphere is proportional to $1/\lambda^4$, where λ is the wavelength, hence the scattering for short wavelength (blue color) is higher than for the long wavelength (red color). The Rayleigh scattering model available in POV-Ray includes the dependency from the direction of the light, but not the dependency from the wavelength, hence here two models are included in the tool: the first is a simple but computationally efficient model that uses that simple Rayleigh scattering with scattering color hand-tuned to give photorealistic result, the second is the high-fidelity model that uses a custom function to restore the dependency also from the wavelength. The high-fidelity model uses the exponential law for the variation of the density of the atmosphere with the height: $\rho = \rho_0 \exp{(-h/H)}$. Where ρ_0 is the density at sea level, h is the altitude of the point considered, and H is the scale height, that here is assumed to be a mean value between 8.4 Km (standard value for the Troposphere) and 6.3 Km (standard value for the Troposphere). A comparison of images rendered with the Earth fully in view both with a simplified and an accurate atmosphere model against a true picture of the Earth from the Apollo 17 mission[22] is provided in Fig. 3. Notice that, although both the synthetic images (Figs. 3b-3c) can not perfectly replicate the true image (Fig. 3a), mostly due to a difference in image resolution, optical

parameters of the true camera not modeled, and different exposure or light conditions, the synthetic images have both a high degree of similarity with the real one. As for the case of the topography mapping, also here it must be considered that the high-fidelity method has an extremely high computational burden with respect to the simpler model and, as it can be noticed from Fig. 3, the results are comparable for altitudes above 1000 Km. For lower altitudes, the results achieved by the low-fidelity model are poorer, mostly in the transition region between the atmosphere and the deep space. The tool uses by default the accurate model for camera altitudes below 1000 Km, otherwise, the default is the simpler model.

(a) True image from Apollo 17 mission (credit NASA).

(b) Image rendered with simplified atmosphere model.

(c) Image rendered with high-fidelity atmosphere model.
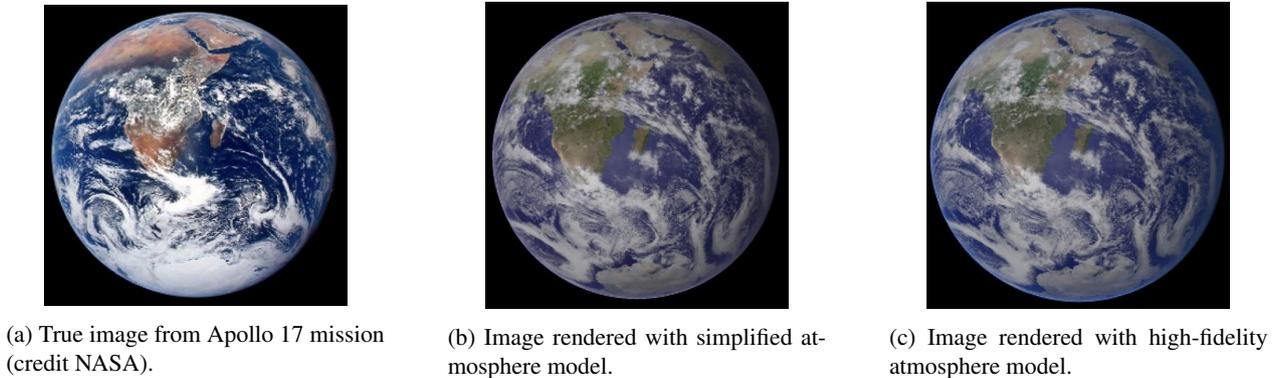
Figure 3: Comparison of images rendered with different atmospheric models against a true Earth picture.

It is remarked that all the models discussed above are available in the tool and they can be selected and combined by the user. It must be noticed that by changing a few parameters in the atmospheric model (i.e. the density model and the wavelength dependency) it is possible to simulate different atmospheres of different celestial bodies. The same holds for the texture and the topography map. Notice that, in contrast to the SPEED and SPEED+ datasets where the Earth is not rendered with the spacecraft but is added in post-processing, here the background is rendered together with the spacecraft, making it possible to directly include in the images the effect of the albedo on the spacecraft itself, without using other light sources to simulate it.

3.2 Spacecraft modeling

The spacecraft (or any other object) can be inserted in the rendered scene by modeling it directly in POV-Ray with primitive shapes. This method is simple but not efficient, especially for spacecraft with complex shapes. The pipeline adopted here is based on directly exporting a POV-Ray compatible file from Blender, as already used in.[11] The input to the entire pipeline is a 3D CAD model representative of the target spacecraft. The target 3D CAD model is exported in Standard Triangle Language (STL) and directly imported into Blender. Notice that, to assign to each surface different optical properties or textures, it is needed or to import the STL generated by a 3D CAD assembly, hence composed by parts bonded in a final assembly, that are interpreted in Blender as children objects, or to split the parent object in children objects in Blender by selecting the faces of the mesh belonging to each child. After that, by using the POV-Ray add-on available for Blender, it is possible to both add custom code to assign textures and optical properties to the objects and to render in Blender the scene with POV-Ray as the rendering engine. Optionally, it is possible to export also the file in Scene Descriptor Language (SDL) used to render the scene. The SDL contains all the information about the textures and the custom code previously added, together with all the children objects, but treated as split objects. To ease working with the exported spacecraft model, the SDL file is hand-modified by merging all the children objects in a single general parent object representative of the entire spacecraft, which is then converted into a POV-Ray include file. Note that all the children's optical properties are preserved also once they are merged into a single parent object. The include file is then used as an external object to be included in the scene description file when needed at the desired location and attitude with respect to the POV-Ray reference frame. This procedure is general and will be tailored to a simplified model of the Tango spacecraft from PRISMA mission in Sec. 4, to validate the tool. The optical properties of the surfaces of the model are tuned during the validation phase.

4. Synthetic Images Validation

To validate the images generated by the proposed tool and to properly tune the optical properties of the spacecraft surface, the SPEED dataset has been used as a reference. The validation of the SPEED dataset with respect to some

true images from PRISMA mission has been carried out by comparing the pixel intensity histograms of synthetic images with respect to the true images.[17] The validation of the images is here carried out by considering only images without background. This constraint for the validation is due to the fact that the information about the rotation of the Earth with respect to the z-axis of the ECI frame and the position of the light source are missing for the SPEED images. Moreover, the clouds in the SPEED images are not reproduced by textures since they are taken from true images of the Earth[17] making not possible to exactly reproduce the SPEED images with background, jeopardizing the validation procedure. Concerning the spacecraft model, the official CAD used for the SPEED dataset is not publicly available, hence a simplified 3D CAD model has been reconstructed from the dimensions of the main components available in:[35] the solar panel is a polygon of 560×750 mm, the main body is a polyhedron of 560×550×300 mm, and all the appendages have a length of 204 mm. Concerning the camera parameters used for the SPEED dataset,[17] they are reported in Tab. 1. These parameters are used to set the perspective pinhole camera model used in POV-Ray to generate the images for validation. Notice that in POV-Ray it is also needed to specify the aspect ratio, equal to

Table 1: Parameters of the camera used for the SPEED dataset.

Parameter	Value
Number of horizontal pixels	1920 px
Number of vertical pixels	1200 px
Focal length	17.6 mm
Pixel length	5.86×10^{-3} mm

AR = 1920/1200 = 1.6 for the SPEED images and the angular field of view of the camera that is computed for a pinhole camera with squared pixels as:

$$\alpha = 2\arctan\left(\frac{CCD_{size}}{2f}\right) \tag{1}$$

Where CCD_{size} is the size of the sensor and f is the focal length. For the SPEED camera $\alpha \approx 35.45°$. To properly recover the relative pose associated with each image of the SPEED dataset, the left-handed reference frame in POV-Ray is shifted to a right-handed reference frame by defining in the camera specifications a *right* vector (the vector used in POV-Ray to describe the direction to the right of the camera) equal to $[-AR, 0, 0]^T$ and the *sky* vector to $[0, 0, 1]^T$. In such a way the POV-Ray global reference frame is made coincident with the ECI frame.[11] The last parameter to be specified to reproduce the images of the SPEED dataset are the location and pointing direction (boresight axis) of the camera, the position of the *arealight* used to approximate the Sun and the position and attitude of the spacecraft. Having available only the relative position between the spacecraft and the camera per each SPEED image, the camera has been placed in the origin of the POV-Ray global reference frame, with pointing direction coincident with the z-axis in the ECI frame, then the relative pose annotated has been used to correctly locate the Tango model in the scene. The position of the Sun has been hand-tuned in order to have shadows in the rendered image as close as possible to the SPEED image considered. Notice that in order to compare the SPEED images with those generated by the proposed tool, the noiseless images are preprocessed in order to add the same noise level as the SPEED images. In agreement with,[17] the images are blurred by applying a Gaussian blurring with $\sigma = 1$ and then a white Gaussian noise with variance $\sigma^2 = 0.0022$ is added to the blurred grayscale image.

Since the size of the spacecraft reported in[35] and used here for the simplified model is not perfectly the one of the Tango spacecraft (which size is not publicly available, up to our knowledge) and since the 3D model is a simplified version that lacks the details of the official mock-up used for the SPEED dataset, the images are firstly preliminarily validated by performing the histogram comparison as in[17] by using the SPEED images as ground truth. Then, in order to have quantitative indexes, the validation is carried out also by considering the shadow index and the feature quality index.[30] The shadow index J_s is computed as:

$$J_s = 1 - \frac{D_s}{S_{real}} \tag{2}$$

Where D_s is the sum of non-zero pixels in the disparity map and S_{real} is the number of pixels classified as "in shadow" in the reference SPEED image. The disparity map is defined by subtracting the binary images obtained by applying the Otsu thresholding[24] to both the tested and the reference image. The threshold level automatically identified to discriminate shadows and illuminated pixels in the SPEED image is used to define S_{real}. As a consequence, the index J_s represents the number of pixels in shadow correctly reproduced in the rendered image. The feature quality index FQI is computed as:

$$FQI = 1 - \frac{\mu(H_d)}{H_{d,max}} \tag{3}$$

Where H_d is the Hamming distance between two corresponding features descriptors extracted in both the images and matched and $H_{d,max}$ is the maximum possible Hamming distance. The mean value $\mu(H_d)$ is computed on 10 matched features extracted using ORB.[34] This index is used since several algorithms rely on feature extraction, hence comparing the features extracted in the reference image with the ones extracted in the synthetic rendered image offers an evaluation of the similarity of the images at the features level. Both the J_s and the FQI are such that the closer to 1, the higher the similarity between images. In agreement with,[30] the validation is considered to be performed successfully if $J_s > 0.90$ and $FQI > 0.80$.

The validation is carried out by replicating 20 random images from the SPEED dataset with the proposed tool. The optical parameters used for the spacecraft model tuned during the validation are reported in Tab. 2.

Table 2: Optical parameters of the spacecraft main components.

Parameter	Solar Panel	Antennas	Main Body
Ambient	0.0001	0.2	0.2
Roughness	0.15	0.0005	0.0005
Brillance	0	3.15	3.15
Diffuse	0.3	0.95	0.99
Reflection	0.25 ÷ 0.5	0.65 ÷ 1.0	0.65 ÷ 1.0
Specular	0.005	0.95	0.95
Phong	0	0.43	0.43
Phong Size	0	0.25	0.25
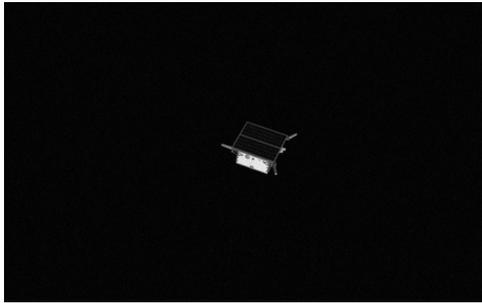
(a) SPEED image *img002350.jpg*.
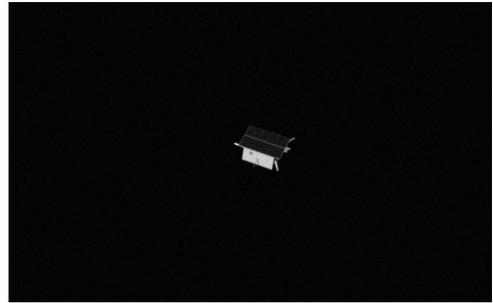
(b) Noised image *002350* rendered with the proposed tool.
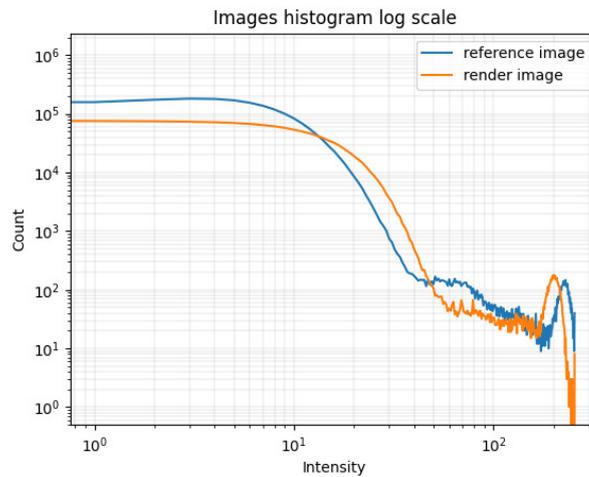
(c) Histogram comparison between 4a (reference) and 4b (render).

Figure 4: Histogram comparison between SPEED image "*img002350.jpg*" and the reproduced noised image.

An example of the comparison of the histograms computed during the validation phase is given in Fig. 4, where Fig. 4a shows the reference SPEED image, Fig. 4b shows the noised image reproduced with the presented tool, and Fig. 4c shows the two histograms compared. By comparing Fig. 4a and Fig. 4b it can be noticed that visually the two images are close, even if it can be noticed that the dimensions given in[17] used here only approximate the true shape of the Tango spacecraft and that the simplified CAD model used in Fig. 4b does not include all the details (e.g. the support structures for the antennas) included in the official CAD. Concerning the histograms in Fig. 4c, it can be noticed that even if they are not perfectly matching, the trends are similar. The differences are mainly given by a bias in the black region, due to a slightly different illumination condition between the two images and a lack of details that are darker in the original SPEED image, this explains why this bias is obtained in all the images. Notice that the light-source position has been hand-tuned by taking the original SPEED images as reference. The J_s and FQI indexes for the 20 random images compared are reported in Tab. 3. The results show that with an average value of $J_s = 0.98$, the rendered images have a shadow distribution extremely close to the reference SPEED images, the same holds for the quality of the features extracted, with an average score of $FQI = 0.83$. Notice that scoring a high FQI means that the ORB descriptors extracted in the two images are similar, hence both the illumination and the material of the original image are correctly reproduced in the rendered image.

Table 3: J_s and FQI indexes evaluated for the tested images.

SPEED image number	$J_s > 0.90$	$FQI > 0.80$
img000012	0.93	0.83
img000043	0.99	0.83
img000059	0.99	0.85
img000114	0.99	0.81
img000138	0.99	0.84
img000324	0.97	0.86
img000388	0.99	0.84
img000468	0.96	0.81
img000486	0.97	0.81
img002122	0.99	0.81
img002156	0.99	0.87
img002350	0.99	0.85
img003271	0.99	0.82
img003849	0.99	0.83
img004386	0.99	0.86
img005964	0.98	0.81
img006967	0.99	0.82
img007408	0.99	0.83
img007461	0.99	0.83
img007485	0.92	0.84
Average value	**0.98**	**0.83**

From the results obtained, since both the indexes satisfy the requirements $J_s > 0.90$ and $FQI > 0.80$ for all the random images considered, the validation campaign can be considered successfully concluded, paving the way to the generation of validated synthetic images datasets.

5. Multi-Purpose Labeled Spacecraft Dataset

The tool described above has been used to produce a set of images of the simplified and validated Tango model with the Earth as background. The parameters of the camera model adopted for the dataset generation are reported in Tab. 4. The camera is modeled from the parameters of the Chameleon 3 by FLIR, with reduced array size. The aspect ratio is equal to 1 and, from Eq. 1, the angular field of view results to be $\alpha \approx 45.55°$. The dataset produced contains about 30000 synthetic noiseless images for training and 3000 images for testing, partitioned with a 10:1 ratio between train and test set. For each split, about 1/6 of the images have the Earth as background, while the others have a perfectly black background. All the images are 16-bit grayscale images in PNG format with 1024×1024 pixels resolution. The images are such that the target spacecraft is located at an altitude ranging from 1500 Km to 6500 Km. The relative distance between the target and the camera is uniformly distributed in a range between 5 m and 30 m. Examples of 10 random images with background extracted from the dataset are reported in Fig. 5.

Table 4: Parameters of the camera used for datasets generation.

Parameter	Value
Number of horizontal pixels	1024 px
Number of vertical pixels	1024 px
Focal length	6 mm
Pixel length	4.8×10^{-3} mm
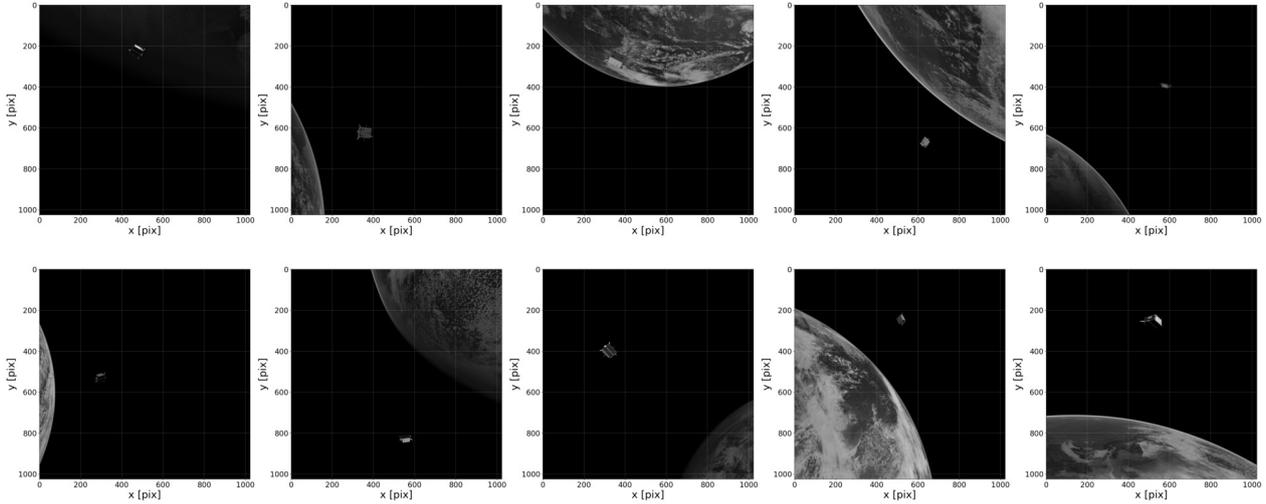
Figure 5: Random images with background from the generated dataset.

The main algorithm places the Earth in the scene at the origin of the ECI frame, after that, the position of the target around the Earth is defined by randomly picking the altitude in the previously defined range, and defining then it is placed by following a uniform distribution around the Earth (see Fig. 6a). The attitude of the target with respect to the ECI frame, named $R_{ECI2TRG}$, is randomly defined and used to correctly place the Tango model in the POV-Ray scene. As it can be noticed from Fig. 6b, the attitude of the target with respect to the ECI frame is uniformly distributed in roll, pitch, and yaw angles.
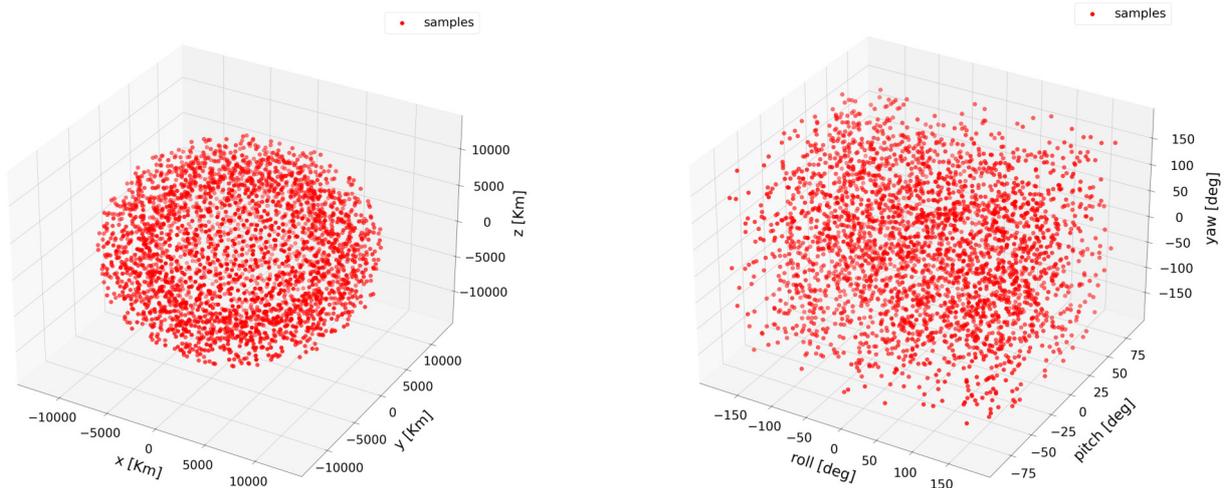
(a) Target position distribution in the test set in ECI reference frame.

(b) Target attitude distribution in the test set with respect to the ECI reference frame.

Figure 6: Target position and attitude distribution with respect to ECI reference frame in test set.

The camera is located in the scene by randomly selecting a position in a sphere centered in the target position with a radius in the range between 5 m and 30 m. To fully define the attitude of the camera with respect to the ECI

frame it is needed to define the pointing direction (i.e. the "*look_at*" parameter in POV-Ray). To avoid having the target always located at the center of the image, the pointing vector has been defined by adding a random displacement vector to the target position vector in the ECI frame. The displacement vector is constrained in order to keep the target inside the field of view of the camera used. The attitude of the camera with respect to the ECI frame $R_{ECI2CAM}$ can be defined per component as:

$$\hat{z}_{ECI2CAM} = \frac{p_{cam} - l_{cam}}{\|p_{cam} - l_{cam}\|} \tag{4}$$

$$\hat{x}_{ECI2CAM} = \frac{\hat{z}_{ECI2CAM} \times s_{cam}}{\|\hat{z}_{ECI2CAM} \times s_{cam}\|} \tag{5}$$

$$\hat{y}_{ECI2CAM} = \frac{\hat{z}_{ECI2CAM} \times \hat{x}_{ECI2CAM}}{\|\hat{z}_{ECI2CAM} \times \hat{x}_{ECI2CAM}\|} \tag{6}$$

Where p_{cam} is the camera pointing vector in ECI frame, l_{cam} is the camera location vector in ECI frame, and $s_{cam} = [1, 0, 0]^T$ is the "*sky*" vector in POV-Ray. Note that now it is possible to define the ground truth relative pose, i.e. the attitude of the target with respect to the camera in camera reference frame $R_{TRG2CAM}$ and the relative position of the target with respect to the camera in camera reference frame $t_{CAM2TRG}$ as:

$$R_{TRG2CAM} = R_{ECI2CAM} R_{ECI2TRG}^T \tag{7}$$

$$t_{CAM2TRG} = R_{ECI2CAM} \left(l_{trg} - l_{cam} \right) \tag{8}$$

Where l_{trg} is the target location vector in the ECI frame. By following the steps formulated above, it is possible to obtain a uniform random distribution of both the target relative attitude with respect to the camera and the target relative position with respect to the camera. Figure 7 shows the relative position distribution (Fig. 7a) and the relative attitude distribution (Fig. 7b) in the test set. Notice that the distribution in Fig. 7a has a conical shape because the target is constrained to be inside the field of view of the camera used.
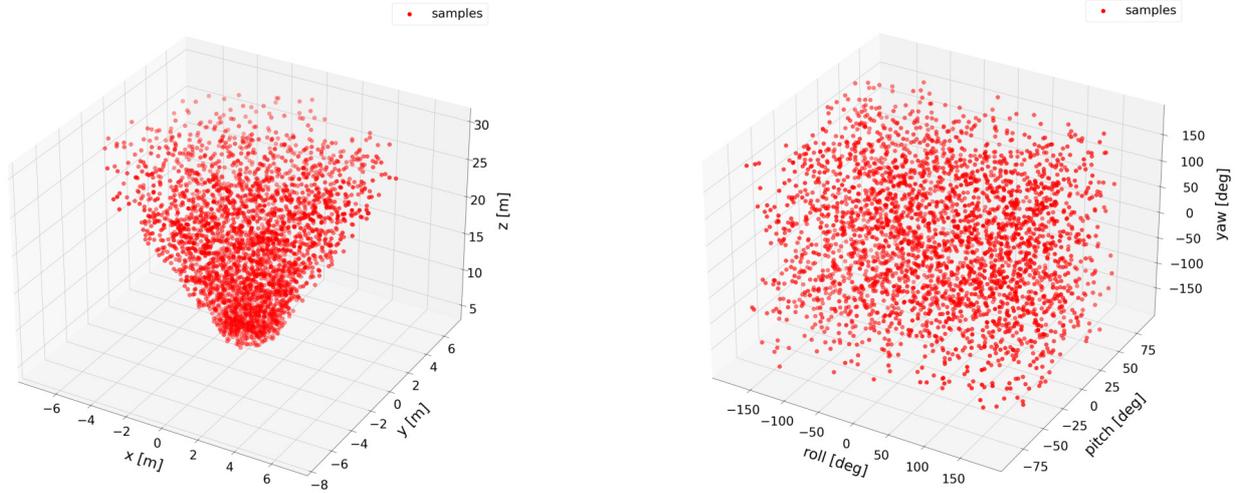
(a) Target relative position distribution in the test set in camera reference frame.

(b) Target relative attitude distribution in the test set with respect to the camera reference frame.

Figure 7: Target relative position and attitude distribution with respect to camera reference frame in test set.

Finally, to properly render the scene, the light source (i.e. the Sun) is placed in the scene on the ecliptic plane by following a random uniform distribution, in order to ensure a complete range of illumination conditions inside the dataset. Notice that the position of the Sun is constrained to be out of the field of view of the camera. Moreover, in order to properly render the image, it is imposed that nor the camera nor the target can be in eclipse. From all the previous information, by exploiting simple geometric relations, it is possible to define before the rendering if the Earth is in the scene and if it is in light or shadow. To speed up the image generation, the Earth is modeled as in Sec. 3.1 only if it is visible and in light, otherwise, it is substituted by an equivalent sphere with equivalent reflection properties in order to always take into account the albedo. Note that this step is extremely advantageous in terms of computational time

because both the parsing time and the rendering time are highly increased if the full Earth model has to be rendered. With the current implementation, the parsing time and rendering time for an image without background are in the order of 0.5 s and 0.4 s respectively, while in the case of visible Earth the parsing time is about 5 s per image, while the rendering time has a high variance, but on average it takes about 2 min per image. The parsing time is evaluated on a single-thread single-CPU process, while the rendering time on a 16-threads 8-CPUs process. The CPUs used are Intel® Core™ i7-9700 CPU, clocked at 3.00 GHz. Notice that the images in Figs. 6-7 are reported only for the test set, but hold also for the train set since it has been obtained with the same tool just by increasing the number of the desired output images. The output pose labels coupled with the images have been collected in a dataset named *Tango Spacecraft Dataset for Monocular Pose Estimation* that has been made publicly available.[1]

The same tool discussed above has been exploited also to create RGB masks for each scene. The RGB images are such that the R-channel (red) corresponds to the spacecraft, the G-channel (green) to the Earth (if present), and the B-channel (blue) to the background (deep space). Per each channel, the pixels have non-zero value only in correspondence of the object that they represent (Tango, Earth, Deep Space). The scene is rendered by adopting a properly tuned single-color texture for both the spacecraft and the total volume occupied by the Earth and the atmosphere. An example of an RGB mask is shown in Fig. 8a. Notice that the RGB masks can be exploited for semantic segmentation tasks. Moreover, by using the spacecraft-related channel, it is possible to extract the associated bounding box for each image. This task has been already accomplished and both the RGB masks and the bounding box annotations have been collected together with the related grayscale images (common to the dataset in [1]) in a dataset that has been made publicly available,[2] named *Tango Spacecraft Dataset for Region of Interest Estimation and Semantic Segmentation*. Concerning the bounding box annotations, they are taken in the image reference frame, with the origin located in the top-left corner of the image. The bounding box is annotated by means of the top-left corner, the bottom-right corner, and the center coordinates in pixels.

To prove the correctness of the relative pose annotations, they have been exploited to project a simplified wireframe model of the Tango spacecraft in the image frame, by using the following transformation:

$$x_{px} = K \left[R_{TRG2CAM} \mid t_{CAM2TRG} \right] x_{TRG}^H \tag{9}$$

Where $x_{px} = [x_{px}, y_{px}, 1]^T$ is the position vector in image frame in pixels of the projected point, K is the camera intrinsic matrix, and $x_{TRG}^H = [x_{TRG}, y_{TRG}, z_{TRG}, 1]^T$ is the position homogeneous vector in target reference frame of a specific point. By using a properly defined simplified mesh of the target spacecraft and exploiting the Möller–Trumbore ray-triangle intersection algorithm,[21] it is possible to extract the line-segment portions that are in view per each image. An example is given in Fig. 8b. Another dataset has been created to include this additional information, hence the same images of the two previously discussed datasets (both in full-scale and in a bounding-box cropped version) have been labeled with the line segments in view per each image. Also this dataset, named *Tango Spacecraft Wireframe Dataset Model for Line Segments Detection*, has been made publicly available.[3]
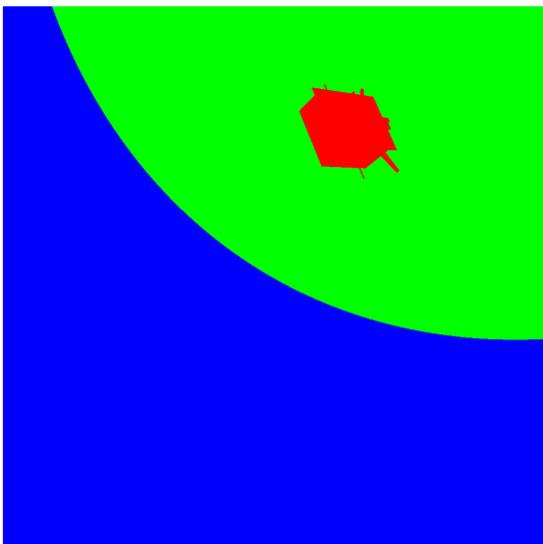
(a) Example of RGB mask image from dataset.
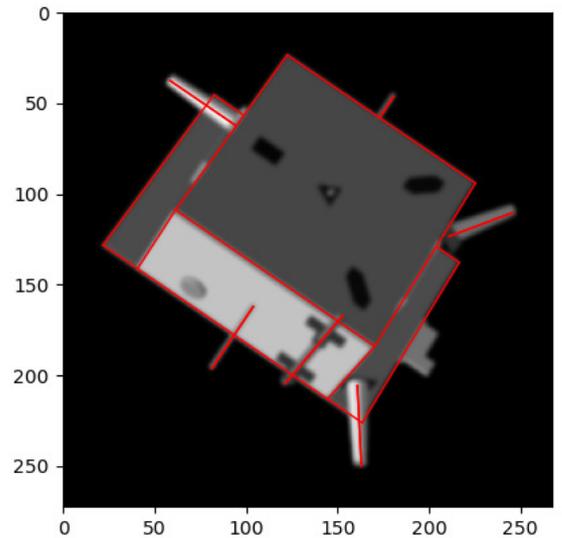
(b) Example of in-view wireframe reprojection in a cropped image of the dataset.

Figure 8: Examples of additional annotations generated.

It is here remarked that these datasets[1–3] contain the same fullscale images, hence they can be used together by combining the annotations of the relative pose, the reprojected wireframe model of Tango, and also the region of interest. These three datasets constitute together the most comprehensive multi-purpose labeled dataset of validated spaceborne synthetic images publicly available, up to our knowledge.

6. Conclusions and Future Works

The paper presented here highlighted the necessity of a tool capable of generating high-quality spaceborne synthetic images to support the development of new algorithms for relative navigation and to serve as a starting point through new missions with a high level of autonomy required. This issue is addressed with a new image generation tool based on POV-Ray, an open-source ray-tracing software, exploited here as the core of the dataset rendering chain. The main assumptions and simplifications adopted for creating representative space-related images in POV-Ray are discussed in detail, leading to models for both the target spacecraft and the background celestial body with a high degree of photo-realism. The simplifications used to model a celestial body with an atmospheric layer and with a well-defined topography have been discussed both in terms of accuracy and rendering time, leading to a trade-off selection of the default functionality implemented in the tool. The implementation for the background object, which can be chosen by the user to be included or not in the scenes, can be used to model several celestial bodies by tuning a few hyperparameters, making the tool proposed highly flexible. Concerning the spacecraft model, here it has been explained in detail the full pipeline from the 3D CAD model to the textured POV-Ray readable include file that can be easily handled by the image generation tool. The pipeline discussed has been applied to the Tango spacecraft from PRISMA mission. The optical properties of the spacecraft have been tuned during the validation phase. The validation of the proposed tool tailored to the generation of spaceborne synthetic noiseless image dataset via physically-based ray-tracing has been carried out both qualitatively via histogram comparison, and quantitatively by evaluating two different indexes with respect to the SPEED dataset images. Despite the assumptions made and the simplified CAD model used, the validation phase highlighted the high level of accuracy of the synthetic images rendered with the proposed tool. In conclusion, the validation phase by using the SPEED dataset as reference has been successfully concluded. The last part of the paper delves into the details of the datasets that have been created by using the previously discussed tool. In particular, the distribution of the relative pose through the datasets is given to show that they are representative of a wide range of scenarios. The relative pose annotations, as well as additional labels as the RGB masks for each scene and the in-view wireframe reprojections in the image reference frame, have been discussed. The datasets created have been made publicly available and it is remarked that since they share the same fullscale images, but with different annotations, they can be combined to constitute the most comprehensive multi-purpose labeled dataset of validated spaceborne synthetic images publicly available, up to our knowledge. It is here remarked that the images are noiseless to increase the flexibility of the datasets since the noises can be easily added in the postprocessing phase to properly tailor the images to the final application scenario.

6.1 Future Works

In the future development, it could be interesting to improve the tool by creating a library of background objects and spacecraft already validated that can be used in a plug-and-play approach. This will ease the process of the creation of new datasets to be used in a wider range of scenarios. To do this it is mandatory to better develop the CAD model of the spacecraft, including also the model of Tango since, as discussed in the paper, the model available is a simplified version. It is here acknowledged that having a highly detailed CAD model is mandatory for some applications such as image segmentation and neural networks training since a strong drift between the train set and the real application scenario, commonly named domain gap, can jeopardize the entire image processing pipeline. Another improvement that can be addressed in future work is the development of algorithms to be included in the tool to better simulate the physics of the light inside a full set of lenses. This will allow us to better simulate real cameras adopted for space missions. Moreover, this improvement in the camera model will make it possible to include algorithms for the evaluation in real-time during the ray-tracing phase of side effects like lens flares or stray light that happens in real-case scenarios and that nowadays are added via dedicated tools in postprocessing without simulating the true physics that brings to these effects.

7. Acknowledgments

References

[1] Michele Bechini, Paolo Lunghi, and Michèle Lavagna. Tango Spacecraft Dataset for Monocular Pose Estimation. *Zenodo*, April 2022.

[2] Michele Bechini, Paolo Lunghi, and Michèle Lavagna. Tango Spacecraft Dataset for Region of Interest Estimation and Semantic Segmentation. *Zenodo*, April 2022.

[3] Michele Bechini, Paolo Lunghi, and Michèle Lavagna. Tango Spacecraft Wireframe Dataset Model for Line Segments Detection. *Zenodo*, March 2022.

[4] Michele Bechini, Marco B. Quadrelli, Michèle Lavagna, and Joseph J. Wang. Hovering of an electrically actuated spacecraft in a small-body plasma field. *Journal of Spacecraft and Rockets*, 58(5):1461–1476, 2021.

[5] Connor Beierle and Simone D'Amico. Variable-magnification optical stimulator for training and validation of spaceborne vision-based navigation. *Journal of Spacecraft and Rockets*, 56(4):1060–1072, 2019.

[6] Roland Brochard, Jérémy Lebreton, Cyril Robin, Keyvan Kanani, Grégory Jonniaux, Aurore Masson, Noela Despré, and Ahmad Berjaoui. Scientific image rendering for space scenes with the surrender software. *arXiv preprint arXiv:1810.01423*, 2018.

[7] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[8] Francesco Castellini, David Antal-Wokes, Ramon Pardo de Santayana, Klaas, and Vantournhout. Far approach optical navigation and comet photometry for the rosetta mission. *Proceedings of 25th International Symposium on Space Flight Dynamics, 25th ISSFD*, 2015.

[9] Bo Chen, Jiewei Cao, Alvaro Parra, and Tat-Jun Chin. Satellite pose estimation with deep landmark regression and nonlinear pose refinement. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 2816–2824. IEEE Computer Society, 2019.

[10] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[11] Francescodario Cuzzocrea. Analysis and validation of spaceborne synthetic imagery using a vision-based pose initialization algorithm for non-cooperative spacecrafts. *MSc Thesis. Politecnico di Milano*, 2020.

[12] Simone D'Amico, Per Bodin, M. Delpech, and Ron Noteborn. Prisma. In Marco D'Errico, editor, *Distributed space missions for earth system monitoring*, chapter 21, page 599–637. Springer Science & Business Media, 2013.

[13] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, jan 1972.

[14] David K. Geller. Orbital rendezvous: When is autonomy required? *Journal of Guidance, Control, and Dynamics*, 30(4):974–981, 2007.

[15] Jacopo Guarneri. Autonomous optical navigation and attitude estimation system tested on artificially generated images. *MSc Thesis. Politecnico di Milano*, 2020.

[16] Brian Karis and Epic Games. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 4(3):1, 2013.

[17] Mate Kisantal, Sumant Sharma, Tae Ha Park, Dario Izzo, Marcus Märtens, and Simone D'Amico. Satellite pose estimation challenge: Dataset, competition design, and results. *IEEE Transactions on Aerospace and Electronic Systems*, 56(5):4083–4098, 2020.

[18] Mate Kisantal, Sumant Sharma, Tae Ha Park, Dario Izzo, Marcus Märtens, and Simone D'Amico. Spacecraft pose estimation dataset (speed). *Zenodo*, February 2019.

[19] Manny R. Leinz, Chih-Tsai Chen, Michael W. Beaven, Thomas P. Weismuller, David L. Caballero, William B. Gaumer, Peter W. Sabasteanski, Peter A. Scott, and Mark A. Lundgren. Orbital Express Autonomous Rendezvous and Capture Sensor System (ARCSS) flight test results. In Richard T. Howard and Pejmun Motaghedi, editors, *Sensors and Systems for Space Applications II*, volume 6958 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 69580A, April 2008.

[20] Paolo Lunghi, Marco Ciarambino, and Michele Lavagna. Vision-based hazard detection with artificial neural networks for autonomous planetary landing. In *13th ESA/ESTEC Symposium on Advanced Space Technologies in Robotics and Automation, ASTRA 2015*, pages 1–8, 2015.

[21] Tomas Möller and Ben Trumbore. Fast, minimum storage ray-triangle intersection. *Journal of Graphics Tools*, 2(1):21–28, 1997.

[22] NASA. Earth From Space - Apollo 17. This image or video was catalogued by Langley Research Center of the United States National Aeronautics and Space Administration (NASA) under Photo ID: EL-1996-00155 and Alternate ID: L86-5815. `https://web.archive.org/web/20080617152411/http://lisar.larc.nasa.gov/UTILS/info.cgi?id=EL-1996-00155`. Accessed: 2022-01-18.

[23] Roberto Opromolla, Giancarmine Fasano, Giancarlo Rufino, and Michele Grassi. A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Progress in Aerospace Sciences*, 93:53–72, 2017.

[24] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[25] Tae Ha Park, Marcus Märtens, Gurvan Lecuyer, Dario Izzo, and Simone D'Amico. Speed+: Next generation dataset for spacecraft pose estimation across domain gap. *arXiv preprint arXiv:2110.03101*, 2021.

[26] Tae Ha Park, Marcus Märtens, Gurvan Lecuyer, Dario Izzo, and Simone D'Amico. Next Generation Spacecraft Pose Estimation Dataset (SPEED+). *Zenodo*, October 2021.

[27] SM Parkes, Iain Martin, Martin Dunstan, and D Matthews. Planet surface simulation with pangu. In *Space ops 2004 conference*, page 389, 2004.

[28] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[29] Massimo Piazza, Michele Maestrini, and Pierluigi Di Lizia. Monocular relative pose estimation pipeline for uncooperative resident space objects. *Journal of Aerospace Information Systems*, 0(0):1–20, 2021.

[30] Margherita Piccinin, Stefano Silvestrini, Giovanni Zanotti, Andrea Brandonisio, Paolo Lunghi, and Michéle Lavagna. Argos: calibrated facility for image based relative navigation technologies on ground verification and testing. In *72nd International Astronautical Congress (IAC 2021), International Astronautical Federation, IAF, Dubai, United Arab Emirates*, 10 2021.

[31] Tomas Plachetka. Pov ray: persistence of vision parallel raytracer. In *Proc. of Spring Conf. on Computer Graphics, Budmerice, Slovakia*, volume 123, page 129, 1998.

[32] Pedro F. Proença and Yang Gao. Deep learning for spacecraft pose estimation from photorealistic rendering. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6007–6013. IEEE, 2020.

[33] Aureliano Rivolta, Paolo Lunghi, and Michele Lavagna. Gnc & robotics for on orbit servicing with simulated vision in the loop. *Acta astronautica*, 162:327–335, 2019.

[34] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. IEEE, 2011.

[35] Sumant Sharma, Jacopo Ventura, and Simone D'Amico. Robust model-based monocular pose initialization for noncooperative spacecraft rendezvous. *Journal of Spacecraft and Rockets*, 55(6):1414–1429, 2018.

[36] Peter Shirley and R Keith Morley. *Realistic ray tracing*. AK Peters, Ltd., 2008.

[37] Shubham Sonawani, Ryan Alimo, Renuad Detry, Daniel Jeong, Andrew Hess, and Heni Ben Amor. Assistive relative pose estimation for on-orbit assembly using convolutional neural networks. In *AIAA Scitech Forum, 2020*, pages 1–11. American Institute of Aeronautics and Astronautics Inc, AIAA, 2020.